

# DACS-USC Data Repository: A Joint Effort by Quanterion and the University of Southern California (USC)

---

*DACS Quanterion Team Member: Thomas McGibbon*

*DACS USC Team Members: Jo Ann Lane, Qi Li, and Ramin Moazeni*

---

## Table of Contents

1.	Introduction .....	1
1.1	DACS-USC Contents.....	1
1.2	Potential DACS-USC Data Sources .....	2
1.3	DACS-USC Data Repository Architecture .....	2
2	Overview of Currently Prototyped Functionalities .....	3
2.1	Data Search and Report.....	3
2.2	Report Formats (TBD).....	10
2.3	Lessons Learned (TBD) .....	10
2.4	Data Submission (TBD).....	12
2.4.1	Preparer information: .....	12
2.4.2	Project description .....	13
2.4.3	Effort/staffing details.....	14
2.4.4	Product size .....	15
2.4.5	COCOMOII Cost Drivers (Optional) .....	16
2.4.6	Quality .....	17
2.4.7	User Management System (TBD) .....	17

## 1. Introduction

The goal of the DACS-USC data repository is to capture and analyze software and software engineering data that will be used to improve:

- Quality of software-intensive systems
- Ability to predict the development of software-intensive systems with respect to effort and schedule.

This repository will:

1. Provide searchable data to support
  - a. Cost estimation: rough order of magnitude estimates<sup>1</sup> based upon
    - Actual data – at least four records with no one organization providing more than 50% of the data points
    - Software size and application domain
  - b. Project planning and management: life cycle model information, key risks, lessons learned, templates, estimation heuristics, software quality trends/data
2. Support software, systems, and system of systems (SoS) engineering research.

### 1.1 DACS-USC Contents

The repository will contain data primarily related to the development of software-intensive systems. This includes data related to the software, software engineering, systems engineering for software-intensive systems, and SoS engineering (SoSE) for net-centric, software-intensive SoS. Overall plans are for the inclusion of the following:

- **Software engineering, systems engineering, and SoSE : Size, schedule**, total cost of ownership, interoperability, and quality data/trends
- **Data entry forms**
- **Administrative data management tools**
- Lessons learned (acquirer and supplier perspectives)
- Counting rules for quantitative data
- Software/system characteristics: Life cycle models, architecture styles, tools used, tool evaluations
- Process-related artifacts: Templates, characteristics, home-grounds

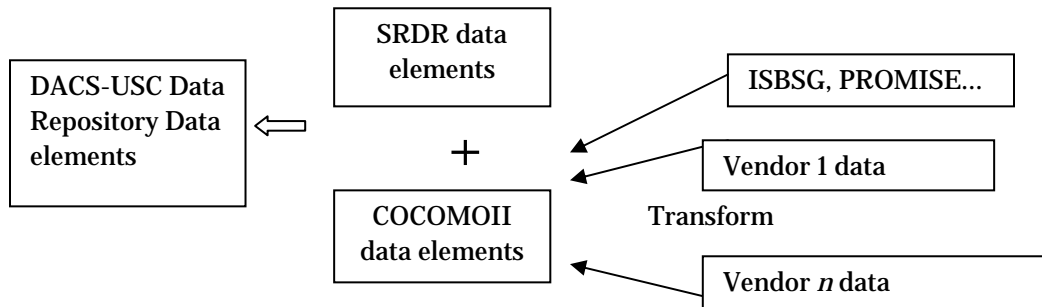
The bolded items are those that are planned for the initial version of the repository.

---

<sup>1</sup> Repository will provide links to participating cost model vendors for more detailed/precise estimates and additional support.

## 1.2 Potential DACS-USC Data Sources

Figure 1 identifies candidate data sources for the repository.



**Figure 1**

The two sources planned for the initial version of the repository are sanitized COCOMO™ projects that are approved for incorporation through USC CSSE agreements with the data owners and data from the Department of Defense (DoD) Software Resources Data Report (SRDR) submissions from software system developers. Other generally available data such as ISBSG and PROMISE are currently being evaluated for inclusion into the repository. In addition, inputs are being solicited from cost model vendors as well as other organizations that have actual data from completed projects. Actual data in repository will depend on ability to find sources for the various data elements.

In general, all data will be stripped of any information that might indicate the original data source, project, or developer and any necessary identifying information needed to maintain the contents of the repository will be kept offline in protected media. (Note: Repository data is only displayed in an aggregated form where there are at least four projects in the aggregation and no one organization has over 50% of the data sets. This is in accordance with the USC Center for Systems and Software Engineering Data Management and Security Procedures provided in Appendix A.)

## 1.3 DACS-USC Data Repository Architecture

The architecture of the DACS data repository will be a 3-tier architecture comprised of open source products to the extent possible. The general structure of the 3 tiers is:

Client Tier: HTML+CSS

Middle Tier: PHP

Backend Tier: MySQL

More detailed information on the repository architecture and security strategies is available in the DACS-USC Data Repository Project Plan that has limited distribution at this time.

## 2 Overview of Currently Prototyped Functionalities

The current DACS-USC DATA REPOSITORY prototype focuses on the Software Data, including functionalities “Searching for similar software projects and show statistics report” , “Import software data” and “display detailed project information” (only for administrator). Capabilities related to system engineering and SoSE data will be implemented in later iterations.

### 2.1 Data Search and Report

After the user logs into the DACS-USC Data Repository (repository) homepage as shown in Figure 2, the left bar provides the instructions on the above “Dear\*\*, Welcome to DACS-USC Repository! Choose Software, System or System of System Data below to find similar projects in this repository, if you want to contribute your organization's data, please go to Contribute Data Section. ” and lists the primary repository feature links. Click on the “Software Data” below “Data Repository” and it will lead to the software data query page in Figure 3.



**Figure 2**

In the software data query page in Figure 3, users can currently query existing software data by “Size” and “Domain” as displayed on top of this page.

# DACS-USC DATA REPOSITORY: Draft Overview

## DACS-USC DATA REPOSITORY

Home

Dear Qi Li, Please query by Size and Domain for Software Data

### Data Repository

♦ Software Data

Size	Application Domain				
+ range	% (SLOC) <a href="#">Scatter chart</a>	Domain <a href="#">Pie chart</a>			
<input type="button" value="Search"/>	<input type="button" value="Clear"/>				
Output Statistics					
	Average	Min	Max	Stdev	Chi
Productivity (SLOC/Hour)					<a href="#">Box-plot chart</a>
Effort (Hours)					<a href="#">Box-plot chart</a>
COCOMO Driver					
PREC: Precedentedness					<a href="#">Pie chart</a>
FLEX: Development Flexibility					<a href="#">Pie chart</a>
RESL: Architecture/Risk Resolution					<a href="#">Pie chart</a>
TEAM: Team Cohesion					<a href="#">Pie chart</a>
PMAT: Process Maturity					<a href="#">Pie chart</a>
RELY: Required Reliability					<a href="#">Pie chart</a>
DATA: Database Size					<a href="#">Pie chart</a>
CPLX: Product Complexity					<a href="#">Pie chart</a>
RUSE: Required Reuse					<a href="#">Pie chart</a>
DOCU: Documentation					<a href="#">Pie chart</a>
TIME: Execution Time Constraint					<a href="#">Pie chart</a>
STOR: Main Storage Constraint					<a href="#">Pie chart</a>
PVOL: Platform Volatility					<a href="#">Pie chart</a>
ACAP: Analyst Capability					<a href="#">Pie chart</a>
PCAP: Programmer Capability					<a href="#">Pie chart</a>
PCON: Personnel Continuity					<a href="#">Pie chart</a>
APEX: Applications Experience					<a href="#">Pie chart</a>
PLEX: Platform Experience					<a href="#">Pie chart</a>
LTEX: Language and Toolset Experience					<a href="#">Pie chart</a>
TOOL: Use of Software Tools					<a href="#">Pie chart</a>
SITE: Multisite Development					<a href="#">Pie chart</a>
SCED: Required Development Schedule					<a href="#">Pie chart</a>
Developing Approaches					
% of code automatically generated					<a href="#">Box-plot chart</a>
% of functionality provided by COTS					<a href="#">Box-plot chart</a>
Number of iterations/increments					<a href="#">Box-plot chart</a>
Languages	<a href="#">Pie chart</a>				
Lifecycle Models	<a href="#">Pie chart</a>				

**Figure 3**

After the user inputs the query conditions, and clicks on “Search”, the statistics results (Average, Min, Max and Standard Deviation) and related charts for all the measures (Productivity, Effort, Schedule, COCOMO drivers and Developing Approaches) are displayed in the table below. If you would like to see the definition of the measure or how it was calculated, you can click on it, and it will provide you a detailed explanation as shown in Figure 4. A repository search example is displayed in Figure 5 to Figure 9.

*Topics for further discussion:*

- *What other query conditions are needed in order to find similar projects in the repository?*
- *What other measures do we need to show? (This also means what other data attributes the data contributor should provide? Which are required, which are optional?).*
- *What other statistics or charts do we need to show the searched result?*

*Other candidate search parameters for similar projects within the repository:*

- 1) *Size with ranges*
- 2) *Domain*
- 3) *Main language used*
- 4) *Development type: new or update*

**DRAFT as of 10 June 2010**

### 5) Cost driver values:

- a. *RELY*
- b. *CPLX*
- c. *SCED*
- d. *Etc...*

#### 3.2.1 Precedentedness (PREC) and Development Flexibility (FLEX)

These two scale factors largely capture the differences between the Organic, Semidetached and Embedded modes of the original COCOMO model [Boehm 1981]. Table 7 reorganizes [Boehm 1981, Table 6.3] to map its project features onto the Precedentedness and Development Flexibility scales. This table can be used as a more in depth explanation for the PREC and FLEX rating scales given in Table 6.

Feature	Very Low	Nominal / High	Extra High
Precedentedness			
Organizational understanding of product objectives	General	Considerable	Thorough
Experience in working with related software systems	Moderate	Considerable	Extensive
Concurrent development of associated new hardware and operational procedures	Extensive	Moderate	Some
Need for innovative data processing architectures, algorithms	Considerable	Some	Minimal
Development Flexibility			
Need for software conformance with pre-established requirements	Full	Considerable	Basic
Need for software conformance with external interface specifications	Full	Considerable	Basic
Premium on early completion	High	Medium	Low

Table 7: Scale Factors Related to COCOMO Development Modes

**Figure 4**

### A searching example:

Before the user searches for information about similar projects in the repository, he can first review general information about the repository contents by clicking “Scatter chart” (beside the Size input) as shown in Figure 5 and “Pie Chart” (beside the Domain input) as shown in Figure 6. In this example, 213 software data records are currently in the repository. The “Overview SLOC Effort Scatter Chart” provides an overview of the 213 projects’ size and effort range in the repository. The “Overview Domain Distribution” Chart provides an overview of the projects’ domain distribution in the repository. As shown in Figure 6, the current software database covers 13 domains and most software projects in the repository belong to the “Business” domain. The Business domain has a total of 45 records, consisting of 21.1% of all software projects.

*Note: We should provide an authoritative list of domain categories. One solution is referring to SEER-SEM’s domain list, other information is USC “AFCAA Database & Metrics Manual” team is now doing some research on this and we can communicate with them about this at ARR.*

# DACS-USC DATA REPOSITORY: Draft Overview

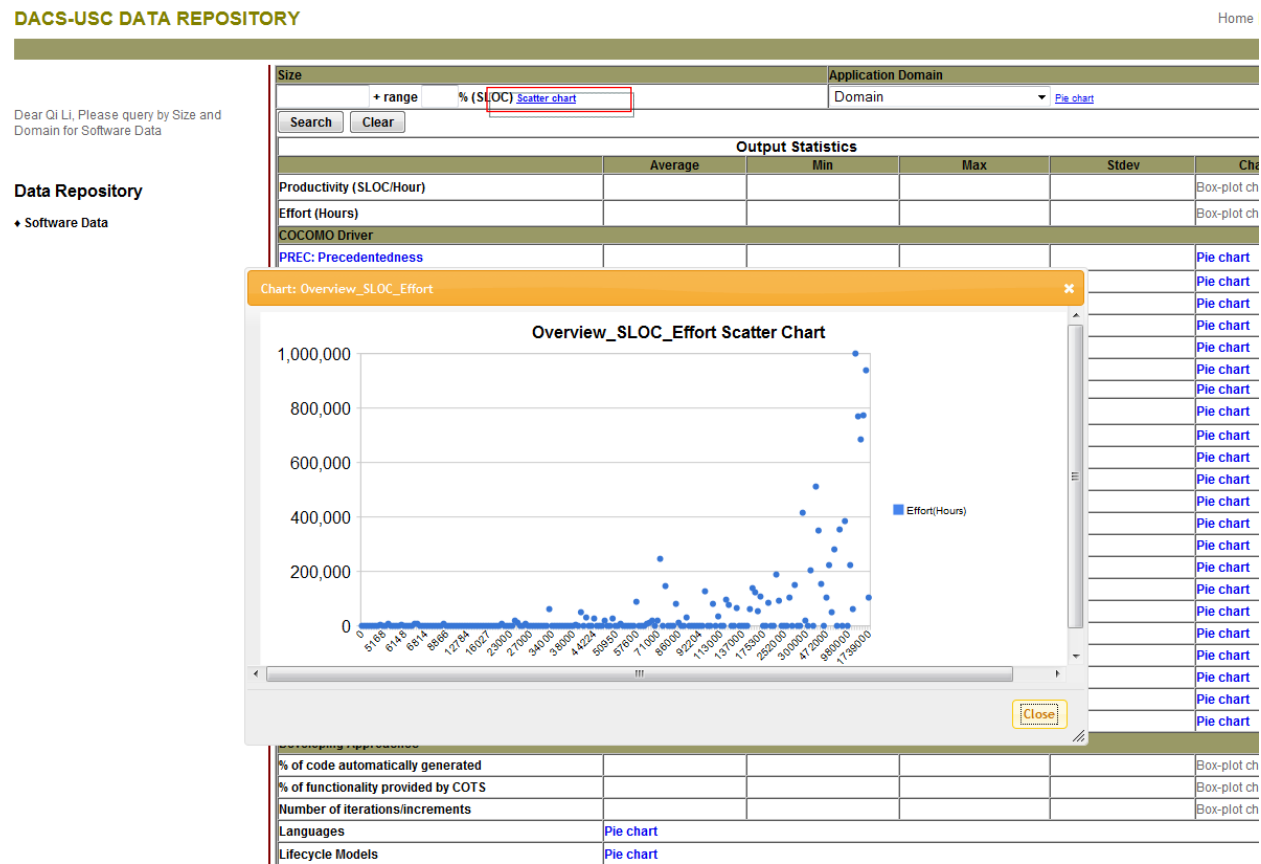


Figure 5



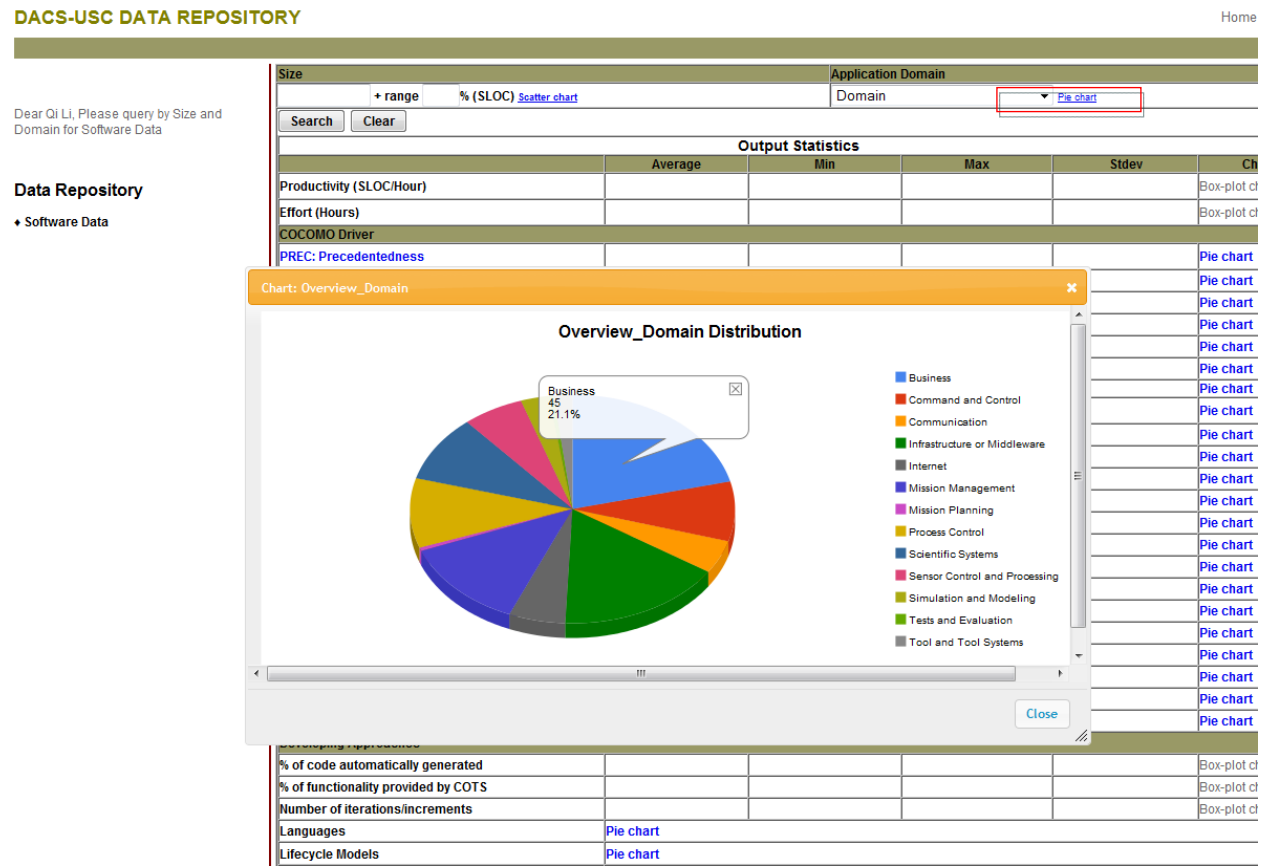


Figure 6

As shown in Figure 7, suppose that the user estimates his current project’s size to be “60000” SLOC, but feels that the estimates have a lot of uncertainty. One option is for the user to widen the search range by 50%, meaning that the system should search for size in the range of  $60000 \times (1-50\%)$  to  $60000 \times (1+50\%)$  or [30000, 90000] SLOC. If the user’s project belongs to the “Business” domain, the user would choose “Business” in the dropdown list.

# DACS-USC DATA REPOSITORY: Draft Overview

## DACS-USC DATA REPOSITORY

Home

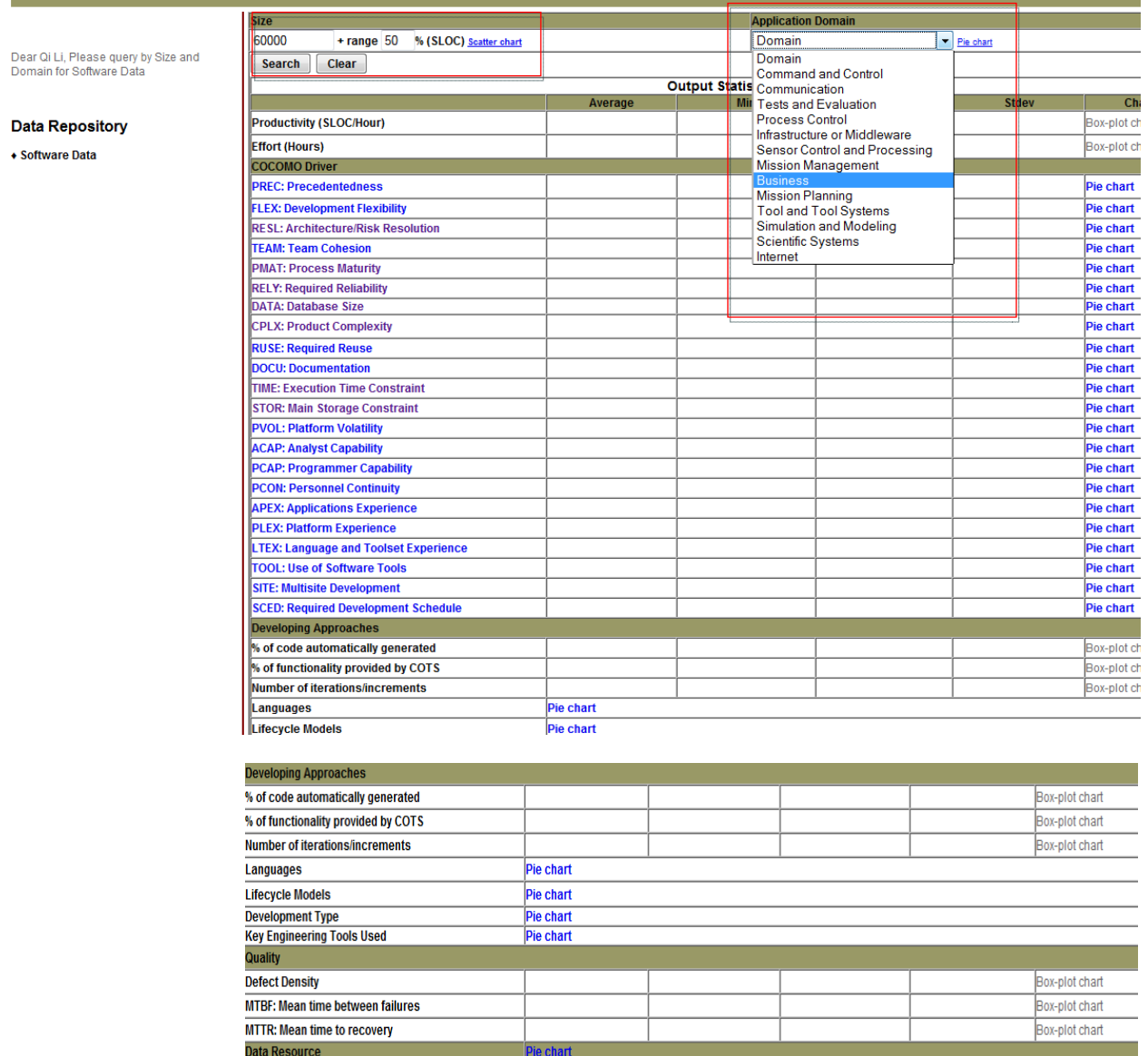


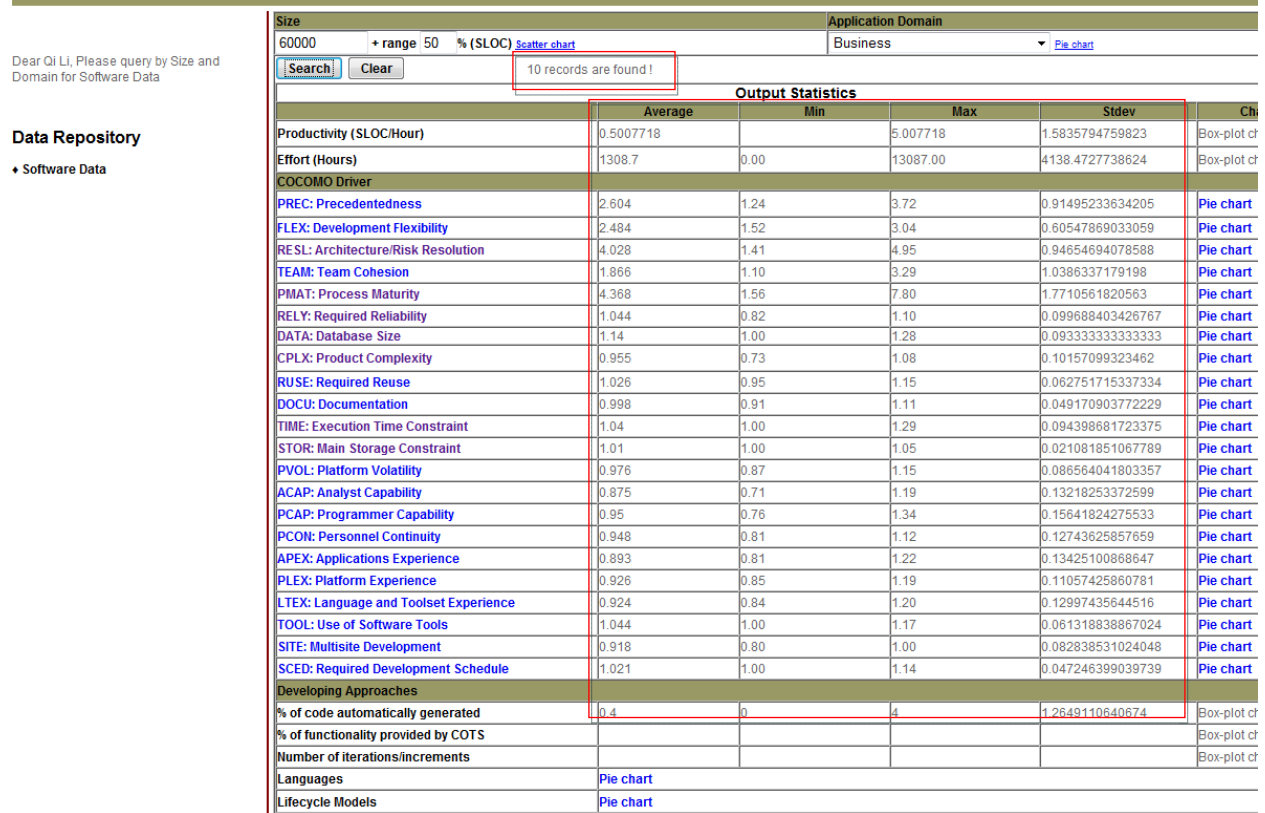
Figure 7

When the user clicks on “Search”, it shows that “10 records are found!” and all the related statistics are displayed on the “Output Statistics” table, as shown in Figure 8. This information provides the user with some useful insights about the general productivity, cost drivers range, and development approaches used by other similar projects. This information can be used to support the user’s project planning and estimating activities.

# DACS-USC DATA REPOSITORY: Draft Overview

## DACS-USC DATA REPOSITORY

Home



**Figure 8**

To show the statistical results graphically, charts are provided by the system. For example, Figure 9 shows the RESL rating distribution for the 10 projects identified by the query. It shows that 8 (80%) projects have an “Nh” rating for RESL. Later on in the project, a Box-plot chart may be implemented to show the statistics for productivity, effort etc.

# DACS-USC DATA REPOSITORY: Draft Overview

## DACS-USC DATA REPOSITORY

Home

Dear Qi Li, Please query by Size and Domain for Software Data

### Data Repository

♦ Software Data

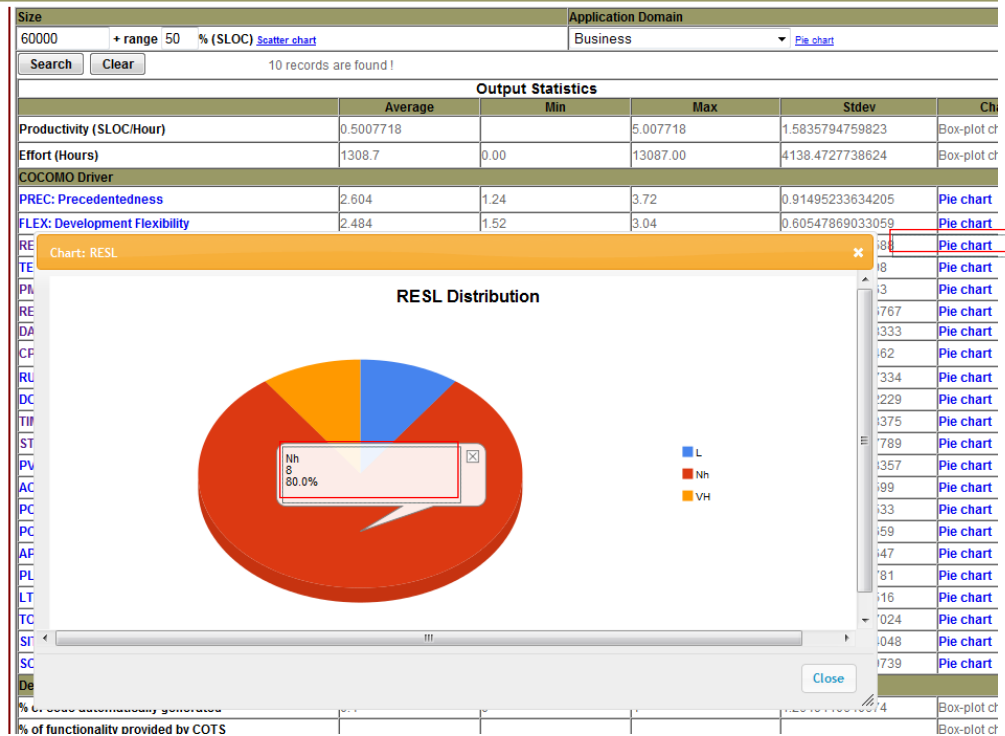


Figure 9

## 2.2 Report Formats (TBD)

This section will describe standard reports that can be requested from the repository. The requested report will be provided in a file that the user can save and print. Exact contents, formats, and file type currently TBD.

## 2.3 Lessons Learned (TBD)

This section will describe the types of lessons learned that will be provided in the repository as well as how the lessons learned will be categorized and tagged.

Current category considerations:

1. Since we can use risks to capture lessons learned, we can also use risk categories for classifying lessons learned. The cost-driver factors in software cost-estimation models provide a checklist of risk items that are strongly correlated with cost and schedule increases on past projects. Please refer to Green Book (Rick Selby edited book) p429-436.
  - a. Aspect of Personnel
  - b. Aspect of Schedule & Budgets
  - c. Aspect of Requirements
  - d. Aspect of External Components and Tasks
  - e. Aspect of Performance
  - f. Aspect of Computer Science Capabilities

2. We can also refer to “*The Macro Risk Model: An Early Warning Tool for Software-Intensive Systems Projects*” (<http://csse.usc.edu/csse/event/2009/UARC/material/Macro Risk Model.pdf>) and add the following categories:
  - a. Aspect of Process Management
  - b. Aspect of Strategies
3. Furthermore, we can consider categories identified in “*A Software Engineering Lessons Learning Repository*” (by Warren Harrison) (<http://citeseerx.ist.psu.edu/viewdoc/download?jsessionid=C01D23197A34A5088F4DDA90DC74E37E?doi=10.1.1.11.1761&rep=rep1&type=pdf>):
  - a. Aspect of Domain
  - b. Aspect of Phase (Requirement->Design->Code->Test->Transition)
  - c. Aspect of Product and Functional Category
4. Other categories to consider:
  - a. Aspect of Risk Management
  - b. Aspect of Quality Management, including Review, Test and V&V etc.
  - c. Aspect of Feasibility Analysis, e.g. Cost-Benefit analysis, ROI analysis, etc.
  - d. Aspect of Project Planning
  - e. Aspect of Effort/Cost/Resources Estimation

### DACS-USC DATA REPOSITORY

[Home](#)

Dear Qi Li, You can learn from these lessons learned information

#### Data Repository

##### ◆ Lessons Learned

#### 1. Systems Engineering on Embedded Weapon System Programs

Systems engineering practices followed on many embedded weapon system development programs are less rigorous and less complete than software engineering practices. The result is that the software development part of system development begins with inconsistent, incomplete, needlessly risky, and highly volatile specific partial list of systems engineering problems observed on more than one program follows.

- No or very inadequate trade-off studies are conducted to reduce the risk of high-risk requirements, particularly high-risk software requirements.
- There is no or negligible participation of software engineers in systems engineering.
- Systems engineering processes and methods are selected indirectly by choosing a systems engineering CASE tool instead of selecting the processes and methods based on the nature of the application, and then selecting the CASE tools that best implement the selected processes and methods.
- There is no modeling and simulation of the system architecture to verify that the architecture will support system requirements for security, performance, safety, and fault tolerance.
- No operational scenarios are developed as part of the system requirements that must be satisfied for system acceptance.
- Interoperability with external systems and compliance with JTA, ATA, etc., is not a central focus in the development of the system architecture.
- Perspective for partitioning the system (e.g., data, states, objects, functions) is not selected in coordination with software engineering for the purpose of minimizing complexity of traceability of system requirements into software architecture.
- There is no systematic and rigorous approach to making requirements consistent.
- Defining requirements that are primarily met through system architecture design (e.g., security, fault tolerance, performance) is delayed until an incremental release after the system architecture has been defined.
- “Evolutionary” design of system architecture greatly increases the risk of excessive rework.

#### 2. Safety and Security

Safety requirements are inadequately flowed down to the software components of the system. Achieving the needed security with the massive, heterogeneous, complex networks that are central to the 21st Century U.S. military is a very difficult technical problem at the leading edge of current technology.

- Software hazard analyses are often not done and are almost never integrated as part of the software engineering process.

**Figure 10 Example of potential lessons learned.**

## 2.4 Data Submission (still under development)

The data submission user input screens are primarily based upon information available from the DoD SRDR forms. Note that in many cases, we have annotated the section of the SRDR form that the indicated data comes from.

Note: SRDR data elements are attached at the end of this document. More information is “AFCAA Database & Metrics Manual” ([http://csse.usc.edu/afcaa/manual\\_draft/](http://csse.usc.edu/afcaa/manual_draft/))---especially DATA ITEM DESCRIPTION.

The screenshot shows a web form titled "Software Engineering Data Submission". It has a tabbed interface with six tabs: "Preparer Info", "Project Description", "Effort/Staffing Details", "Product Size", "COCOMOII Cost Drivers (Optional)", and "Quality". The "Preparer Info" tab is selected and highlighted. Below the tabs, there are several input fields for preparer information: "Prepared Date:", "Preparer First Name:", "Preparer Last Name:", "Preparer Email Address:", "Preparer Contact Address:", "Preparer Organization:", "Preparer Work Phone:", and "Preparer FAX NO:". Each field is represented by a text box. At the bottom right of the form, there are two buttons: "Close" and "Save".

**Figure 11 data elements categories**

### 2.4.1 Preparer information:

This screenshot is similar to Figure 11, showing the "Software Engineering Data Submission" form with the "Preparer Info" tab selected. However, a red rectangular box is drawn over the right side of the form, containing the text "Section 3.6". This box highlights the area corresponding to the "COCOMOII Cost Drivers (Optional)" and "Quality" tabs, indicating that the data in these sections is derived from Section 3.6 of the AFCAA Database & Metrics Manual.

**Figure 12 preparer information**

## 2.4.2 Project description

Software Engineering Data Submission

Preparer Info   **Project Description**   Effort/Staffing Details   Product Size   COCOMOII Cost Drivers (Optional)   Quality

Project Name:

Year of Start:

Year of Completion:

Number of Iterations/Increments:

Functional Description:  **Section 3.2.1**

Operating Environment:  or add new:

Application Domain:  or add new:

Application Type:  or add new:  **Section 3.2.3**

Development Type:  **Section 3.2.3.4**

Development Process:  or add new:  **Section 3.2.3.5**

Development Method:  or add new:  **Section 3.1.9**

Software Process Maturity:  or add new:  **Section 3.2.3.1**

Development Languages: add language

Language Name	Language %
add COTS/GOTS	
COTS/GOTS Name	Integration Effort (Hours)
add tools	
Tools Name	Used for

COTS/GOTS Application Used:  **Section 3.2.4**

Key Engineering Tools Used:

**Figure 13 project description**

## 2.4.3 Effort/staffing details

Software Engineering Data Submission											
Preparer Info		Project Description		Effort/Staffing Details		Product Size		COCOMOII Cost Drivers (Optional)		Quality	
Requirement Analysis (Hours):	Section 3.4.1	<input type="text"/>	Section 3.4.4	Start Month	<input type="text"/>	Section 3.4.4	End Month	<input type="text"/>			
Preliminary Design (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Detail Design (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Code & Unit Test (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Integration (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Qualification Testing (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Development Test and Evaluation (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Maintenance (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Other (Hours):		<input type="text"/>		Start Month	<input type="text"/>		End Month	<input type="text"/>			
Total (Hours):		<input type="text"/>									
Hours/PM:	<input type="text"/>										
Software Effort Comments:		Section 3.4.5									
<hr/>											
Peak Staff:	<input type="text"/>	Section 3.2.5									
Peak Staff Date:	<input type="text"/>										
Hours per Staff-Month:	<input type="text"/>										
<b>Personel Experience in Domain:</b>											
Highly Experienced (%):	<input type="text"/>	Section 3.2.6									
Normalal Experienced (%):	<input type="text"/>										
Inexperienced/entry level (%):	<input type="text"/>										

**Figure 14 effort/staffing details**



## 2.4.4 Product size

Preparer Info	Project Description	Effort/Staffing Details	Product Size	COCOMOII Cost Drivers (Optional)	Quality
Number of Total Requirements:		<input type="text"/>	Section 3.3.1		
Number of New Requirements:		<input type="text"/>			
Number of Total External Interface Requirements:		<input type="text"/>	Section 3.3.2		
Number of New External Interface Requirements:		<input type="text"/>			
Requirements Volatility:		<input type="text"/>	Section 3.3.3		
Amount of delivered code developed new		Human Generated	Section 3.3.4	<input type="text"/>	
		Auto Generated		<input type="text"/>	
Amount of delivered code reused from external source (i.e. not inherited from previous increment/build or predecessors)		With Modification		<input type="text"/>	
		Without Modification		<input type="text"/>	
Amount of delivered code inherited (i.e. reused from previous increment/build or predecessors)		With Modification		<input type="text"/>	
		Without Modification	<input type="text"/>		
Total Delivered Code		<input type="text"/>			
Comments about Size:		Section 3.3.5			
<b>Using COCOMOII to Calculate Software Size (Optional)</b>					
Code Breakage %:		<input type="text"/>			
Unadjusted Function Points:		<input type="text"/>			
New SLOC:		<input type="text"/>			
Code Count Type for New SLOC:		<input type="text"/>			
Adapted SLOC:		<input type="text"/>			
Code Count Type for Adapted SLOC:		<input type="text"/>			
DM, CM, IM:		<input type="text"/>	<input type="text"/>	<input type="text"/>	
SU, AA, UNFM:		<input type="text"/>	<input type="text"/>	<input type="text"/>	
Software Reuse Comments:		<input type="text"/>			
Total SLOC:		<input type="text"/>			

Figure 15 product size

## 2.4.5 COCOMOII Cost Drivers (Optional)

Software Engineering Data Submission

Preparer Info	Project Description	Effort/Staffing Details	Product Size	COCOMOII Cost Drivers (Optional)	Quality
---------------	---------------------	-------------------------	--------------	----------------------------------	---------

Scale Factors:

PREC Rating:  with offset:

FLEX Rating:  with offset:

RESL Rating:  with offset:

TEAM Rating:  with offset:

PMAT Source:

PMAT Rating:  with offset:

Cost Multipliers:

RELY Rating:  with offset:

DATA Rating:  with offset:

RUSE Rating:  with offset:

DOCU Rating:  with offset:

CPLX Rating:  with offset:

TIME Rating:  with offset:

STOR Rating:  with offset:

PVOL Rating:  with offset:

ACAP Rating:  with offset:

PCAP Rating:  with offset:

AEXP Rating:  with offset:

PEXP Rating:  with offset:

LTEX Rating:  with offset:

PCON Rating:  with offset:

TOOL Rating:  with offset:

SITE Rating:  with offset:

SCED Rating:  with offset:

**Figure 16 COCOMOII Cost Drivers (Optional)**

### 2.4.6 Quality

Software Engineering Data Submission

Preparer Info	Project Description	Effort/Staffing Details	Product Size	COCOMOII Cost Drivers (Optional)	Quality
Number of Defects Discovered:		<input type="text"/>			
Number of Defects Removed:		<input type="text"/>			
Defect Density:		<input type="text"/>			
MTBF: Mean time between failures:		<input type="text"/>			
MTTR: Mean time to recovery:		<input type="text"/>			
MTTD: Required or actual mean time to serious or critical defect at delivery in hours		<input type="text"/>		Section 3.5.1	
Observed or computed reliability compared with nominal reliability of analogous systems		<input type="text"/>		Section 3.5.2	
Comments about Quality:				Section 3.5.3	

**Figure 17 Quality Data from SRDR Form.**

### 2.4.7 User Management System (TBD)

*Will address various roles and relevant privileges (administrator, general users, do we have other roles, such as guest etc...)*

# Appendix A: USC Center for Systems and Software Engineering Data Management and Security Procedures

---

# USC Center for Systems and Software Engineering Data Management and Security Procedures

---

This document details the procedures we, at the USC-Center for Systems and Software Engineering (CSSE), take to manage the software/systems project-related data we receive from our affiliates and clients. This agreement applies to all USC-CSSE data collection projects where the data may be considered sensitive or proprietary by affiliate or client data providers/submitters. It currently covers the COCOMO™ model suite of data and the DACS-USC Repository data. The procedures are covered as follows:

1. Data Identification
2. Data Submission
3. Data Storage
4. Data Access
5. Getting Help

## 1. Data Identification

Each Affiliate software project contributing data has a separate file identification number of the form XXX-YYY-NN-yyyymmdd.

XXX is one of a random set of three-digit organization identification numbers provided by USC to the Affiliates and only known to the Affiliate and Barry Boehm or Jo Ann Lane at USC CSSE. Note that a given Affiliate organization may have more than one identifier. For example, an Affiliate may request multiple IDs to distinguish various organizational sites or internal organizations.

YYY is a three-character alpha-numeric ID number assigned by the Affiliate when establishing a new project data file. Only the submitting Affiliate knows the correspondence between YYY and the actual project name.

NN is a two-digit number to distinguish multiple data submissions from a single project corresponding to different development cycles or iterations.

yyyymmdd identifies the date the data was submitted to the repository.

## **2. Data Submission**

Data submission is an important aspect of this procedure. There is a need for data submission with the protection to the Affiliates' privacy. There is also a need to be flexible in delivery medium. In addition, the procedure used must be possible to detect unusual or erratic data and trace it back to its source.

### **2.1 Protection Level and Duration**

Each submitter (identified by XXX) is asked to provide a "sunset" time at which point the data is no longer requires strong protection and sanitized versions (i.e., source/project identifying information not included) may be made available to others (e.g., the research community and USC CSSE clients and associates). Submitters can indicate a number of years or "none". Submitters should specify "none" for data they do not ever want made available to anyone outside of those on the USC CSSE data access list. If a "sunset" time is not provided by the submitter or other representative from his/her organization, it will be assumed to be 10 years from the date of submission. If a submitting organization gets acquired by another organization, the data will be protected in accordance with the acquiring organization's policy or for 10 years if there is no acquiring organization policy on file.

### **2.2 Format**

The USC CSSE data collection projects would like the data submitted by using the Data Collection form provided or as COCOMO™ data files. Actuals can be captured in a spreadsheet or in a calibration file that is part of the COCOMO™ package. Alternatively, data can be submitted in other formats given that the format has been reviewed and approved by a member of the data access team.

## **3. Data Storage**

The USC CSSE data collection project has obtained a secured room with limited access at USC for its data facility. The door is outfitted with a cypher lock on the door and is locked at all times. Only those personnel on the authorized list may enter the room and access the data. Each project's sanitized data<sup>2</sup> is maintained on at least two independent, portable hard drives that are stored in a locked filing cabinet in the secure room when not in use. At least two copies of the data allows for use by multiple authorized personnel as well as providing a reliable backup of the data.

Data submissions received via Internet electronic mail will be inaccessible once it is received and stored on the appropriate project hard drive.

## **4. Data Access**

Access to actual data submissions/raw data is limited to researchers on the project whose names appear on the USC CSSE data access list. Researchers will sign nondisclosure agreements for Affiliates when requested by the Affiliate. The raw data will not be made available to Affiliates. Summary data will be made available only if data from at least four Affiliates is included in a

---

<sup>2</sup> Source/project identifying information excluded and only identification number described above used to index/identify data set.

data summary category. In addition, any summary data will not be dominated by a single source (e.g., over 50% of the data from one source). For those cases where there is a dominate source, data records will be randomly selected from the dominant source so that the summary criteria are met. The one exception to these rules is that the COCOMO™ project will provide each Affiliate with summaries of their own data.

For sanitized data records that have reached their “sunset time”, records are incorporated in an online repository. Summary data is made available through this online repository in response to user queries that result in data from at least four data sources. This data is made accessible via username/password protections so that access can be controlled and monitored.

## **5. Getting Help**

### **5.1 Phone Help Resources**

#### **5.1.1 USC Affiliate Questions**

Direct questions on affiliate matters, meeting dates and locations, copies of proceedings, and general administrative matters to these people:

Julie Sanchez (jasanche@usc.edu).....(213) 740-5703  
Center for Software Engineering FAX.....(213) 740-4927

#### **5.1.2 Questions About Data**

For general questions or questions on the USC COCOMO™ model, data definitions, or project data collection and management, contact:

Barry Boehm..... (213) 740-8163  
Jo Ann Lane..... (858) 945-0099  
Center for Software Engineering FAX..... (213) 740-4927  
Internet Electronic-Mail.....cocomo-info@sunset.usc.edu

For questions on the DACS-USC Repository, data definitions, or project data collection and management, contact:

Jo Ann Lane..... (858) 945-0099

## **USC CSSE Controlled Data Access List**

Personnel currently on the access list include:

1. Dr. Barry Boehm
2. Dr. Jo Ann Lane
3. Dr. Brad Clark
4. Thomas Tan
5. Qi Li
6. Ramin Moazeni